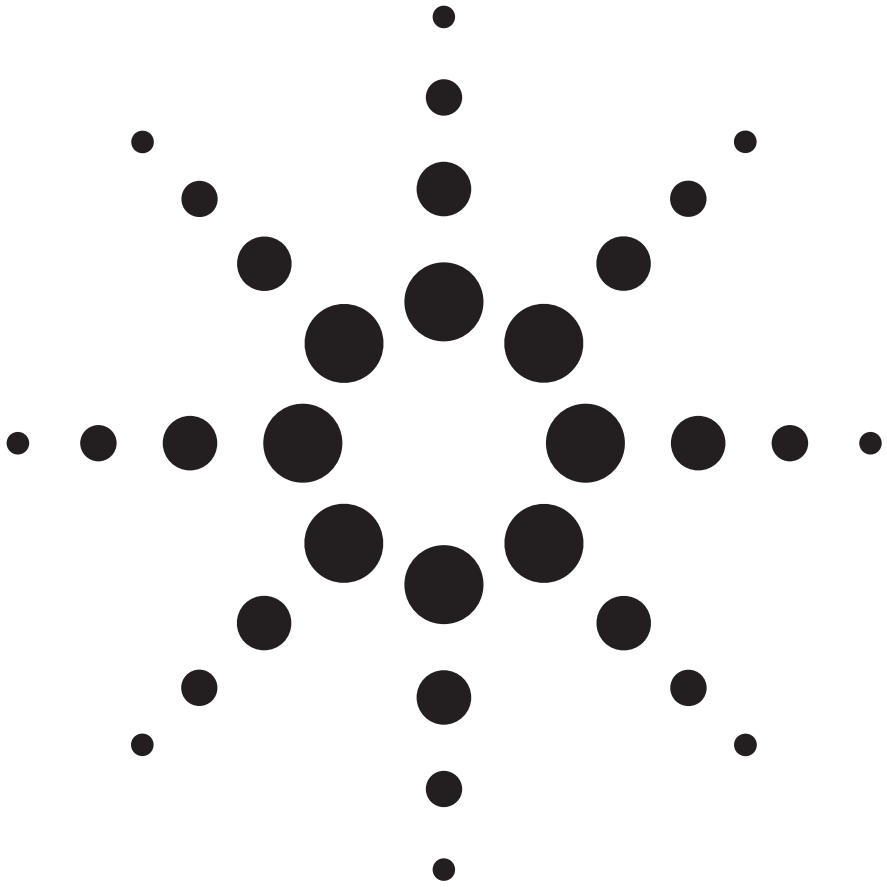

Release Notes for Agilent TestExec SL 5.1

Agilent Technologies

Agilent TestExec SL 5.0 - 20 December, 2001

TABLE OF CONTENTS FOR THE MAJOR SECTIONS IN THIS FILE

Section 1. GENERAL NOTES

A. Installation Notes

If you have any TestExec SL 5.0 beta release installed, you must uninstall it before installing this release.

There have been some installation problems if a Microsoft Office application (or anything that uses Visual Basic for Applications) is running during the installation. We recommend that you exit all running applications prior to the installation (you may have to reboot anyway).

If you have TestExec SL 4.1 installed, the installation program will allow you to either upgrade your existing installation or co-install TestExec SL 5.0. In the case of an upgrade, TestExec SL 4.1 will be removed from your system (unless another system is still referencing it) and your existing user files directory (the TestExec SL 4.1 default was "\My TxSL Files") will be reused and its "Preferences.upf" file updated. If you choose to co-install, the default for the user files directory is "\My TxSL 5.0 Files". You can change this, but the installation program will prevent you from naming it the same as an existing TestExec SL 4.1 user files directory.

If you are co-installing this release with TestExec SL 4.1, you should copy files "TxSLSSField.fdef", "TxSLSSRecord.rdef", "TxSLXMLField.fdef" and "TxSLXMLRecord.rdef" from directory "C:\Program Files\Agilent\TestExec SL 5.0\DefaultConfiguration" to the "Test Systems" subdirectory of the TestExec SL 4.1 user files directory (typically "C:\My TxSL Files\Test Systems") to overwrite the old version's datalogging definition files.

This installation can coexist with a TestExec SL 4.1 installation with one exception: the TestExec SL Action Wizard 4.1. If you need to use the old Action Wizard, do the following:

1. Close all TestExec SL processes.

2. From either a DOS command prompt or the "Start | Run..." menu, execute "regsvr32 C:\Program Files\Agilent\TestExec SL 4.1\bin\ txsleditors.ocx".

3. Run the TestExec SL Action Wizard 4.1. At this point, TestExec SL 4.1 can be run but TestExec SL 5.0 cannot.

4. Close all TestExec SL processes.

5. From either a DOS command window or the "Start | Run..." menu, execute "regsvr32 C:\Program Files\Agilent\TestExec SL 5.0\bin\ txsleditors.ocx".

B. To run Agilent TestExec SL, launch "Agilent TestExec SL 5.0" from the Start menu in the Windows taskbar.

C. To view the Release Notes for this software release, do the following:

1. Start online help by choosing "Contents" from the Help menu in TestExec SL's menu bar.

2. When the introductory topic titled "Welcome to Agilent TestExec SL" appears, click the "Release Notes" link adjacent to "See Also" at the bottom of the topic.

Section 2. CHANGES FROM 4.1.1 TO 5.0

A. This release has Microsoft Visual Basic for Applications (VBA) built into it as a language you can use to create actions. See the online Release Notes for a link to more information about this feature.

B. TestExec SL now supports the Agilent Fault Detective. If the Fault Detective is installed prior to TestExec SL 5.0, the installation process will make the appropriate customizations to your "Preferences.upf" file. If you install Fault Detective after TestExec SL, you must either manually make those customizations or reinstall TestExec SL. See the notes in file "\Program Files\Agilent\TestExec SL 5.0\Examples\FaultDetective\readme.txt".

C. The behavior of the TestExec SL sequencer has been changed so that choosing "Ignore All Failures" only affects when the testplan stops and does not affect the judgment of the testplan.

In previous release of TestExec SL, the "Ignore All Failures" setting affected both the running of the testplan (it would run until the end of the testplan) and affected the judgment (all testplans were marked as passing).

If you wish to have the old behavior, where "Ignore All Failures" affects both the running of the Testplan and the judgment, then you should add the following entry to the "Preferences.upf" file.

```
[Process]
;This entry is optional, and if not present, the default is No.
Ignore Failures Forces Pass = Yes
```

D. This release supports the checking of multiple pass/fail limits per test. See the online Release Notes for a link to more information about this feature.

E. The setup|cleanup|execute model for actions has been simplified to an execute|cleanup model in this release. See the online Release Notes for a link to more information about this feature.

F. This release supports the use of COM (Component Object Model) objects. You can define a COM object as a module in TestExec SL's topology and then access it from action code. See the online Release Notes for a link to more information about this feature.

G. This release lets you examine the actual values of parameters in parameter blocks while a testplan is paused. See the online Release Notes for a link to more information about this feature.

H. TestExec SL has a free evaluation period that lasts thirty days. After that, TestExec SL will quit operating and you must redeem your license if you wish to continue using TestExec SL. See the online Release Notes for a link to more information about licensing.

KNOWN DEFECTS IN THE 5.0 RELEASE

A. If you unload a VBA project upon which the currently loaded testplan relies, TestExec SL will not automatically reload it when trying to run the testplan. You must either unload/reload the test plan or reload the VBA project.

B. Bug #SGDU00007710: First VBA user dialog displays behind TypicalOpUI.

   The first time a testplan containing a dialog box or user form invoked by a VBA action is run from TypicalOpUI (or any other Visual Basic operator interface), the dialog box or form appears behind the TypicalOpUI form.

   WORKAROUND: Click the "Agilent TestExecSL" button in the Windows taskbar to bring the dialog box or form to the front, and then proceed normally. After this, the dialog box/form will correctly appear in front of the TypicalOpUI form.

C. The default path when saving a VBA action project should be "$TestplanDir$\Bin" if a testplan is loaded or "$UserFilesDir$\Test Systems" if no testplan is loaded. This does not always work correctly. If you choose the Save button in the VBA Action Project Management box the path defaults to the first "Bin" directory found, which may not be the one you want. The workaround is to explicitly specify -- i.e., use Save As -- where to save the project file.

D. The profiler will not display its results if the testplan fails. Even if you set the Sequencer Halting option (Options | Testplan Options | Execution tab) to Ignore All Failures, TestExec SL still reports a testplan as failing

if any of its tests fail. To view profiler results in a testplan with failing tests, you must skip the failing tests (select one or more tests and choose Debug | Skip).

E. When using the editor for data whose type is "Range" it is possible to enter an illegal step size without generating an error message. If you enter an illegal step size, it will be changed automatically to a legal value when you click the OK button for the dialog box.

F. Your project will not link if you include spaces in the names of instruments when using the Action Wizard. You can work around this by enclosing the name of an instrument in double quotes where it is referenced in the Project | Settings | Link | Input... Object/Library Modules field in Visual Studio.

G. If you have developed a custom operator interface in Visual Basic, it will generate an unrecoverable error under the following conditions:
- If the VB application attempts to access any non-password related TestExec SL method before validating a successful login that happened either manually or via TestExec SL's automatic login feature
- If TestExec SL is either in its evaluation period and warning of its expiration or at the end of the licensed period and warning of the remaining days until expiration.

The workaround is to insert the appropriate snippet of code into the operator interface code prior to any other accessing of TestExec SL:

If using the automatic login feature...

```
While Not TestExecSL1.Security.IsAUserLoggedIn
Wend
```

If not using the automatic login feature...

```
While Not TestExecSL1.Security.IsAUserLoggedIn
TestExecSL1.Security.Login "ValidUserName", "ValidPassword"
Wend
```

Section 3. CHANGES FROM VERSION 4.1 TO 4.1.1

A. A change to 'seq.dll' creates a new sequence called 'AfterMain' in any testplan loaded in TestExecSL 4.1.1. The 'AfterMain' sequence is only added to testplans that do not have it, and its presence does not change the behavior of old testplans. However, users can make use of this sequence to perform operations like marking good UUT's after testing in the 'Main' sequence completes normally.

Section 4. CHANGES FROM VERSION 4.0 TO 4.1

A. TestExec SL is now a commercially available product.

B. References to HP or Hewlett-Packard in TestExec SL and its documentation have been changed to Agilent or Agilent Technologies.

C. TestExec SL now has a Web site at
   http://www.agilent.com/find/testexec where you can find
   up-to-date product news. Also, an Action Wizard to help
   you create actions in Microsoft Visual C++ will soon
   appear there for downloading.

D. This release includes two tutorials to get you started using
   TestExec SL. You can access them under Tutorials in TestExec
   SL's Help menu.

E. All of TestExec SL's documentation is now online. This makes
   it easier to find information because the index of help topics
   now spans all topics. Also, you can do text searches across all
   topics to find items of interest, plus add topics of interests
   to a list of Favorites for easy reuse. For more information,
   see "Using Online Help" in the online help.

   Note about printing from online help: You have the option of
   printing a selected topic or all topics beneath a given point
   in the hierarchy of help topics. A bug in Microsoft's HTML Help
   viewer causes some topics to print with incorrect formatting
   when you use the "Print the selected heading and all subtopics"
   option. If this happens, you must print the affected topics
   individually.

F. This release adds an automatic login feature that lets you skip
   the login process if security is not an issue for you. This
   feature is enabled by default but it can be disabled. For more
   information, see "Logging In Automatically" in the online help.

G. Previous releases of TestExec SL stored user preferences and
   other transient data in an initialization file named
   "tstexcsl.ini". This release replaces the initialization file
   with a "preferences file" whose name is "preferences.upf". The
   contents of this file are described in various topics in online
   help as needed.

H. Besides running under Windows NT Service Pack 5 or later with
   Internet Explorer 4 Service Pack 1 or later, this release runs
   under Windows 2000.

I. This release adds a generic switching handler that lets you
   control various switching modules whose programming is
   message-based. It is readily customizable to support
   additional switching hardware. For more information, see
   "Working with the Generic Message-Based Switching Handler"
   in the online help.

J. This release adds a new sequence named AfterMain. Items you
   place in it are executed after the Main sequence has run to
   completion. For more information, see "Branching on a Passing
    or Failing Testplan" in the online help.

K. This release changes the hierarchy of TestExec SL's directories
   to improve the distinction between system areas used by TestExec
   SL and areas that contain files created and maintained by users.
   In particular, this change makes it easier for users to migrate
   testplans and their associated code from development to production

systems. For more information, see "Standard Directories" in the online help.

Note: Because TestExec SL's directory structure has changed, installing the new software on an existing system will not remove or replace the old software. Be aware that any shortcuts pointing to the old software will continue to point to the old software, which may cause problems unless you correct the shortcuts.

L. This release of TestExec SL is not an update for the TestExec SL software that comes with Agilent TS-5400 and TS-5500 test systems. Those systems come with their own versions of TestExec SL. Because this version of TestExec SL and the TS-5400/5500 versions of TestExec SL are installed in different locations, this version and TS-5400/5500 versions can reside on the same system.

M. The configuration files for TxSL style, spreadsheet-compatible datalogging have been modified to include the LimitArrayIndex field. This allows for easier interpretation of tests that return arrays of values.

N. TestExec SL now comes with a more extensive set of predefined actions for your use. These include general-purpose timing actions plus actions for controlling specific instruments:

   Agilent 34401 DVM
   Agilent 54600 Oscilloscope
   Agilent 33120 Arbitrary Waveform Generator
   Agilent 3499A Switch/Control Unit
   Agilent 6613C Power Supply

   For more information, see "Predefined Actions Provided with TestExec SL" in the online help.

   Also, example actions to control the Agilent 34401A DVM are provided for:

   HP BASIC for Windows
   Agilent VEE
   Microsoft Visual C++
   National Instruments LabView

   For more information, see "Examples of Actions in Multiple Languages" in the online help.

O. This release adds an option that lets you set the location of the system topology file from the development environment.

P. When you create a new testplan TestExec SL now prompts you with a default name and location for the testplan. The location is automatically added to the search paths for the testplan.

Q. TestExec SL now supports the use of shortcut notation in the form of self-expanding macros that substitute for longer pathnames. TestExec SL sometimes inserts these macros into pathnames, such as in dialog boxes, or you can manually type them as shortcuts. For more information, see "Shortcut Notation in the File System" in the online help.

R. The evaluation version of TestExec SL is identical to the full version except that the evaluation version quits running after thirty days.

Known bugs for the 4.1 release.

A. Because of incompatibilities between program code linked with National Instruments LabWindows/CVI and program code linked with Microsoft Visual Studio 6.0, you will encounter difficulties using LabWindows/CVI program code with Agilent TestExec SL. Please contact Agilent technical support for more information and assistance.

Section 5. CHANGES FROM VERSION 3.22 TO 4.0

A. This release adds a feature called the Throughput Multiplier that lets you test multiple UUTs with a single set of hardware resources and a single testplan. Because of reduced UUT handling time and better overlapping of tests, this reduces the test time per UUT and increases the utilization of your test system. Also included is an example of an operator interface written in Visual Basic that supports multi-UUT testing. For more information, see Chapter 10 in the printed "Using Agilent TestExec SL" book.

B. This release adds a feature that lets those who are familiar with the commands used to control message-based instruments create easy-to-use actions to control instruments for those who are not familiar with the commands. For more information, see Chapter 9 in the printed "Using Agilent TestExec SL" book.

C. This release adds a feature that lets you create strings that contain replaceable parameters. For more information, see Chapter 8 in the printed "Using Agilent TestExec SL" book.

D. This release includes redesigned data editors used to edit the characteristics of parameters and symbols. The redesigned data editors provide a tree view on the forms in which they appear plus a more detailed view on multiple tabs in a separate dialog box.

   Other changes to the editors include:
   * Many data types support units
   * You can view integer data in decimal, hexadecimal, and octal formats
   * The integer data type supports enumeration
   * You can view string data in ASCII, hexadecimal, and octal formats
   * Multi-value data types, such as arrays, support a spreadsheet-style editor that allows cutting and pasting from commercial spreadsheets
   * Multi-value data types support the insertion of comma-separated values
   * You can graphically view data for the real array, waveform and point array data types

For more information, see "Agilent TestExec SL Data Editors" in the table of contents for Agilent TestExec SL's online help.

E. Inserting actions into tests has been simplified. Now you can:
   * Quickly insert an action by typing part of its name and letting Agilent TestExec SL suggest likely choices.
   * Find actions via two levels of searching
   * Select multiple actions to be inserted instead of inserting them one at a time.

F. This release adds a feature that validates the integrity of testplans as you develop them. For more information, see "Testplan Validation Features" in Chapter 3 of the printed "Getting Started" book.

G. As of this release, Agilent TestExec SL no longer is supported on Windows 95. It now requires Windows NT 4 Service Pack 4 or later and Microsoft Internet Explorer 4 Service Pack 1 or later. Also, the code base has been upgraded to run using Version 6 of Microsoft Visual Basic and Microsoft Visual C++. Users who develop code using the Agilent TestExec SL libraries or samples should upgrade to these new versions.

H. Use of the equivalence limit checker for real variables in new code is now prohibited. This change resulted from errors resulting from machine accuracy when attempting to compare real numbers. The high/low or nominal tolerance limit checkers are recommended for use with real numbers.

I. The addition of hex/octal formats and units results in the listing output changing slightly. Users who have written utilities that parse the listing output may need to modify these utilities.

J. The system requirements for running Agilent TestExec SL are now:

   * IBM-compatible PC. A PentiumÆ Pro 200 MHz (or equivalent) with 64 MB of RAM is the minimum, and a PentiumÆ II 300 MHz or faster with 128 MB of RAM is recommended.

   * CD-ROM drive

   * 1024 x 768 graphics or better

   * At least 100 MB of free hard disk space. A minimal installation of Agilent TestExec SL requires about 32 MB, and a full installation requires about 35 MB. The remaining space is needed for testplans, action definitions, and other files that you create with Agilent TestExec SL.

   * Microsoft Windows NT 4.0 with Service Pack 4 or later

   * Microsoft Internet Explorer 4 with Service Pack 1 or later

Known Bugs for 4.0 release:

A. If you are running the developer's environment and then

simultaneously run the operator interface written in Visual Basic, the two environments will be linked together. This is an unsupported mode that can cause unpredictable behavior when editing the testplan while in this state. If you decide to use this mode you should save often and avoid making major changes to the testplan. Operating in this mode will also result in double reporting output being sent to the Visual Basic operator interface, and to datalogging files (if the logging of reports is enabled.)

B. You should not use dashes (-) preceded by spaces or followed by spaces in the names of variants. Variant names that use spaces with dashes can cause Agilent TestExec SL to lock up.

C. If you double-click a testplan in Windows Explorer to open it in Agilent TestExec SL, the testplan name will appear in the short 8.3 file format in datalogging records. Because this is not normally done from an operator interface or a production environment, it should be noticeable only on a development system.

D. The "<Agilent TestExec SL home>\samples" directory is not properly cleaned up when Agilent TestExec SL in uninstalled. You must manually remove this directory.

E. The Stop button does not work when single-stepping through actions. Instead, you must use the Abort button.

F. The Agilent TestExec SL ActiveX control that lets Agilent TestExec SL interact with code written in Visual Basic is not properly registered until you have run the Agilent TestExec SL development environment at least once. Thus, you should run the development environment at least once before attempting to run any operator interfaces written in Visual Basic.

G. Breakpoints do not work when using the "run selected tests" option (Debug | Run Selected Tests). The workaround is to single-step through the actions in a test (Debug | Set Action Step) and press the Continue button as needed.

H. The results from Agilent TestExec SL's profiler do not appear when testplans fail. The workaround for profiling testplans that fail is to set the "ignore all failures" feature (Options | Testplan Options | Ignore all failures) so that the testplan passes.

I. Setting a "skip" flag (Debug | Set Skip) on "if..then..else" statements" in a testplan does not work properly. The "if" part of the testplan is executed and the "else" part is not.

J. If you using the Agilent TestExec SL ActiveX control in an operator interface written in Visual Basic and attempt to write to a nonexistent symbol in a symbol table, the write operation appear to complete but the symbol will not be written nor will an error be generated. Be sure to use valid names when writing to symbol tables from Visual Basic applications.

K. Agilent TestExec SL's profiler prints only the first page of its

results and not successive pages. The workaround is to view the data online, which displays it in its entirety, use the Windows clipboard to copy sections of the results to another application, such as WordPad, and print from there, or examine the results in Microsoft Excel.

L. When using looping (Options | Testplan Options | Loop for count) in a multi-UUT testplan, the Report window shows the pass/fail counts for only the most recent iteration of the testplan. For example, if the loop count is set to 2 and the testplan contains 10 passing tests, a single-UUT testplan will report 20 passing tests and a multi-UUT testplan will report 10 passing tests for each UUT position. The problem is that the multi-UUT report only reflects results from the last iteration of the loop executed, and not the cumulative results from all iterations.

M. If you are using Agilent TestExec SL with an Agilent TS-5400-family test system, be aware that the test system's I/O bypass mode does not work with the Agilent 33120 arbitrary waveform generator or the Agilent 53131 counter.

N. In some cases the listing for tests (View | Listing | Tests) does not show the full test name. Instead, the name is truncated; e.g., "Test test NewTest1" may appear as "Test tes". Other than making the names of tests somewhat more difficult to determine, this causes no problems.

O. If you are using Agilent TestExec SL with an Agilent TS-5400-family test system, be aware that the test system's Action Wizard will not work properly if you install the Agilent TS-5400 software before installing Microsoft Visual C++. The fix is to reinstall the Agilent TS-5400 software after installing Visual C++ or install Visual C++ before installing the Agilent TS-5400 software.

P. After you make changes to an external symbol table, a dialog box appears that lets you save the changes to the file in which the external symbol table resides. Even if you say "no," the changes are saved to the copy of the symbol table in memory even if you choose the OK button when exiting the symbol table editor. The Save to File dialog box is your only chance to save the symbol table to a file; if you say "no" to it, the next time the testplan is loaded the old symbol table values will be loaded from the old file.

Q. Contrary to what Agilent TestExec SL's documentation states, file "tstexcsl.ini" is removed when you uninstall Agilent TestExec SL. If this initialization file contains changes that you wish to retain, you should copy it to another location before uninstalling Agilent TestExec SL, uninstall and reinstall Agilent TestExec SL, and then copy the old initialization file over the new one.

L. String formatting applied in a testplan to a parameter designated as an Action Output (the parameter's name appears in bold in a list of parameters) is not saved when the testplan is saved. There are two workarounds if you need to save string formatting applied in a testplan for a parameter that is an

action output -- i.e., a result:

* You can reference the output parameter to a symbol in symbol table TestStepParms and do the string formatting there. The string formatting will then be saved with the testplan but it will not be possible to reference other action parameters in the format.

* You can make a copy of the action definition, specify string formatting for the parameter in the copy of the action definition, and use the copy of the action definition whenever it is necessary for the parameter to have string formatting applied to it.

M. Under some circumstances, you may encounter an Invalid Access error when exiting Agilent TestExec SL. This is typically caused by an invalid parameter for a hardware module in the system topology file, "system.ust". If you see this problem, verify that these parameters are correct. Because all of Agilent TestExec SL's files are written to disk before this problem occurs, your chances of losing data are minimal.

N. The Path Editor will not correct an error when you type the name of a switching path containing characters whose case does not exactly match an existing switching path and then attempt to make a correction via the drop-down list of node names without first deleting the incorrect node name. For example, suppose you type an incorrect node name in the editor, such as one that contains "a" instead of "A". If you attempt to fix this mistake by choosing the correct value from the drop-down list without deleting the incorrect node name, the problem appears to be fixed. However, when you leave the current field the node name reverts back to the incorrect value. Two workarounds are:

* You can delete the mistyped node name and use the drop-down list to choose the correct one.

* You can directly edit the path by retyping the correct node name to replace the incorrect one.

Notes About Using Expressions in Flow Control Statements:

Many of the flow of control statements -- e.g., if...then, for...step -- and the assignment statement (=) allow one or more expressions for fields in them. Some suggestions for using expressions in flow control statements are:

* To replace an expression with a reference to a symbol in a symbol table, highlight the entire expression, click the mouse's right button, and assign the reference.

* For expressions where the order of evaluation is important, use parentheses to force that order.

* Be sure that the expression in a field is valid before leaving the field. Valid expressions:
  ** Cannot be blank

        ** Cannot end in an operator
        ** Contain matching sets of parentheses (when parentheses
           are present)
        ** Contain symbols whose names begin with an alpha
           character (not a number) and contain only alphanumeric
           characters
        ** Are case-sensitive

   * When you specify a symbol in an expression and that symbol
     does not already exist in a public symbol table, a new symbol
     whose type is real is automatically created in the
     SequenceLocals symbol table. If the expression does not seem
     to return the correct value, verify that a new symbol of the
     correct name was created in the SequenceLocals symbol table.

Notes About Reloading Changed .umd, .sym, and .ust Files:

   After a testplan is loaded, changes to action definition
   ("*.umd"), external symbol table ("*.sym"), and switching
   topology ("*.ust") files are not necessarily updated
   immediately. This happens because copies of these files are
   held in memory and are not always updated upon making changes
   to them. The workaround in all cases is to exit and reload the
   testplan. Other, more selective workarounds include:

   * You can reload "*.umd" or "*.sym" files by choosing
     Options | Testplan Options. Then choose the Search Paths tab.
     Choose Action Definitions or Symbol Tables from the drop-down
     list associated with "Search paths for." Click on the list of
     testplan-specific files and choose the Apply button to reload
     the files.

   * You can reload "*.ust" files by choosing Options | Topology
     Files and choosing the OK button.

==
Section 6. CHANGES FROM VERSION 3.21 TO 3.22
==

A. Modifying and rerunning a testplan that contained comments and
   statements that relied on a testname (eg. using OnFailGoto to
   branch to a non default statement) would result in a memory
   leak. The memory leak was repaired in version 3.22

==
Section 7. CHANGES FROM VERSION 3.2 TO 3.21
==

A. Nesting of loops is now allowed up to 32 deep. The previous
   limit was 5.

B. Repaired the login screen for the TypicalOpui operator
   interface. It now correctly hides the password.

==
Section 8. CHANGES FROM VERSION 3.11 TO 3.2
==

A. The datalogging system was significantly enhanced. These enhancements include:
   --Support for a new (TxSL) naming scheme that conforms much more closely to TxSL naming conventions. It is much easier to understand. The old (3070) naming scheme continues to be supported.
   --Documentation on the behavior of the data log system.
   --The creation of a TxSL DataLog Configuration editor that allows much easier editing of the configuration files that specify data log operation.
   --Support for an XML data log file format. The XML file format is self descriptive and easily understood, at the expense of being larger. Commercial tools exist that can browse an XML formatted files.
   --Added an ActiveX control that takes an XML formatted TxSL data log output and populates an object model of the TxSL data log results. The intent of this object model is for users to walk the object model, extracting data for insertion into a specific data base format. The object model could also be used to extract information for custom reports.
   --Added new standard capabilities to specify file names, file name prefixes and file name suffixes. Users specify these choices with user defined messages.
   --Support of a new style of data log configuration file. These new files have extensions of rdef (as in record definition) and fdef (as in field definition). Old style files (with .ini extensions) continue to be supported.
   --Added new sample configuration files that specify how to use the new TxSL naming schemes when logging to XML and spreadsheet files. The new configuration files will be loaded in the TxSL\bin directory.
   --In the interest of backward compatibility, did NOT change the tstexcsl.ini file which specifies which data log configuration files will be used. To use new/different configuration files, you should modify the Data Log section of the tstexcsl.ini file.
   --Added support for the use of symbolic references in data logging.
   --Added two user defined messages to the data log system. The AfterDataLogCreateDone message is generated when the data log system has finished processing the data. Included with this message is a single string that represents the data log results. The AfterDataLogFileWriteDone message is generated after the datalog file is written to disc. Included with this message is a string that represents the file name.
   --Added the ability to NOT produce a file. This mode of operation is specified by sending a UserDefinedMessage to TxSL. It is most useful when an operator interface is analysing the data log results returned from with the AfterDataLogCreateDone message, and no physical file is needed.

B. With TxSL3.2, some of the samples are installed via the use of an archive. This results in the brief display of an unarchive window, during the run of the install.

C. The help for the TxSL Development Environment now spans help for all of the TxSL components. For example, using help from the development environment lets you access help topics for the Agilent TestExec Control, the Agilent TestExecSL DataLogging control, and the DataLogging Configuration Editor. Using help from any of the components continues to only give access to help for that component.

Section 9. CHANGES FROM VERSION 3.1 TO 3.11

A. The meaning of the #Items field in the data logging store has been clarified. As in the past, the value of this field depends on the type and size of the data type value.

For scalars (single values, like reals, ints and strings), the value returned was 2, it is now 1.

For single dimension arrays, the value was N, where N is the size of the array. The value remains (unchanged) as N.

The multiple dimension arrays, the value was 2. The new value is minus 1 (-1).

B. The timing of when the TestDesignator field of the data store is initialized to "" was changed. The field is now initialized prior to running tests.

This now allows the field to be overwritten in customer actions. Overwritting the field is NOT recommended. This change was only done to preserve compatibility to a major customers code.

C. You may have more than four loop statements in a testplan. This was a bug introduced in 3.10.

Section 10. CHANGES FROM VERSION 3.0 TO 3.1

Note: Changes from 3.0 to 3.1 did not make the printed manuals. For documentation on any of the below features, please refer to the online documentation.

A. TxSL now has the concept of a broken statement. A broken statement is defined as being a statement with an invalid expression. For example, a statement of "If !bogus then" would be flagged as broken.

Note: TxSL does allow the user to execute a testplan with broken statements, however the broken statements WILL be ignored by the sequencer. This could cause the improper execution order of test statements. Please see the online documentation for more information.

B. The I data types (i.e. IUtaInt32) now raise exceptions when initialized with an invalid name for the parameter. The user may leave this exception handling out by inserting:

```
#define UTA_IGNORE_PARM_NOT_FOUND_EXCEPTION_ON_ICLASS TRUE
```

in their code before including the uta.h header file.

Note: Agilent strongly recommends that the user leaves exception handling enabled for the I data types, since they usually signal a mistake between action definitions and the action source code.

It is not necessary to recompile existing actions. Functionality will not change. The I data types will only raise exceptions if your actions are recompiled without the #defined just mentioned.

C. The concept of running selected statements outside of sequence order has been added. This option allows the user to select and

execute tests outside of the sequencing order. Please see the online documentation for more information.

D. Added a new find dialog so that users can do full and partial word searches, with case sensitivity, on both statements and action names.

E. Datalogging of the Nominal Tolerance limits checker was incorrect. The datalogger used to report the nominal & tolerance values instead of evaluating them and showing the High/Low values. The datalogger now reports the High/Low values for all limits checkers.

F. Fixed a defect where HaltOnFailure and IgnoreFailures were not working properly from the ActiveX control interface. The new behavior is setting either property to true results in the opposite property being set to false. If both properties are set to true, TxSL will ignore all failures.

G. Fixed a defect where once a testplan was aborted, you could no longer single step the testplan. It is now possible for a testplan to be aborted, and then single step a testplan from the beginning.

H. Changes to lister output for switching:

Finding particular setup::cleanup switching behavior can be difficult, and today involves stepping through each test, or a manual examination of the lister output. Since setup::cleanup's that are configured as connect::connect are rare, it is valuable to quickly find these setups. The format of the listings of the testplans and tests has been modified so that it will be easier to find particular switching configurations for setup and cleanup.

The previous behavior of the listing was to show setup::cleanup behavior in the following format:

```
Switching: [wire2]
           [wire2]
           At Test Setup: Connect Paths
           At Test Cleanup: Connect Paths
```

The new behavior of the listing will be to show setup::cleanup behavior in the following format:

```
Switching: [wire2]
           [wire2]
           Setup::Cleanup = Connect::Connect
```

By putting all the setup-cleanup information on one line, it is now simple to search the listing output for configurations of interest. For example, to find all switching actions that have setup::cleanup configured as connect::connect, you would search for the following.

```
           Connect::Connect
```

in the lister output.

I. Repaired major defects in automate.c (a sample), dealing with the use 5430 IO, and the Response IO. Removed the DriveTime parameter from the parm block definition for DirectIO.
This feature had never been implemented.

J. The format of the output parameters when viewing a listing of actions has changed. In earlier releases, the parameter data was printed with no delimiters (other than spaces). This made it difficult to spot the boundaries between parameters, especially when a parameter name or value contained spaces. The format of parameter data when using action listings has now changed so that a vertical bar is used to delimit the boundaries between fields. An example of the new format is shown below.

Parameters:

Parm Name	Value	Type	Description
Input	0	Int32	The input value
Result	-5	Int32	The return value

K. A defect in the lister that would lock up the program when the lister window was resized horizontally has been repaired.

L. A defect in the lister that would GPF (General Protection Fault) upon saving to a file that was already in use was repaired.

M. It is possible to set 3070 style data logging so that information sent to the report window is also embedded in the data log file. In previous versions of TxSL, the report strings that were at the end of the report output would incorrectly occur early in the data log file. This behavior has been repaired, so that report strings that are at the end of the report output also occur at the end of the data log output.

This was accomplished by changing the first parameter on the report line in hpfmtdef.ini from 4 to 2. If backward compatibility is desired, then editing hpfmtdef.ini so that this parameter continues to be equal to 4 is possible. This file is usually found in the Agilent TestExec SL/bin directory.

This change does not impact spread sheet style data logging, unless a custom .ini file that includes report information has been created.

N. The loglevel field (item 29) that is output in a datalogging file has been changed slightly. Loglevels set to "log failures" were previously indicated by the word "fail" in the loglevel field. The new behavior is for this to be indicated by "failures". Loglevels set to "none" were previously indicated by "missing" in the log level field. The new behavior is for this to be indicated by "none".

This change impacts 3070 style data logging, when the standard data logging .ini files are used. It could impact spreadsheet data logging if a custom .ini file has been created that uses this field.

O. If the "ignore all failures" flag is set, the UUT will be considered to have passed, even if individual tests fail. This is because TxSL starts with the concept that a UUT is good, and may be proven bad via any individual test failures. When failures are ignored, the UUT will be never considered to have failed.

With the above in mind, it is possible to get the following report output when "ignore all failures" is set.

```
-------- End of Testplan
 3 Passed Tests
 3 Failed Tests

 UUT PASSED
```

Ignore all failures is best used when you desire the testplan to run to completion, regardless of the passing/ failing of any tests.

P. The cut/copy/paste of tests with variants from one testplan to another was clarified. Tests (and their variants) are pasted into new testplans according to the following rules.

--If the destination testplan has a variant with the same name as a variant in the source testplan, then the tests are simply copied over, and variant information is preserved.

--If the source testplan has a variant that does NOT exist in the destination testplan, then the information for that variant for the pasted tests is lost. In other words, a new variant is NOT created in the destination testplan.

--If the destination testplan has a variant that does NOT exist in the source testplan, then as a test is copied from source to destination, new variant information for the copied test is created, and initialized with values from the first variant of the source testplan. In normal usage, this is the normal variant.

Q. Repaired a defect in the ui_debug.dll sample code that would, on approximately the 50th cycle of loading, running and unloading a testplan, result in an abnormal termination of TxSL.

R. The datalogging ".ini" files were modified so that item 50 and 51 are swapped. They are now in the correct position. This error was not visible in previous versions because the incorrect positions in both the data store and format files cancelled each other out. In this release, they are correct. Users should see no change in behavior.

The default value for item item 52 in dsdef.ini and ssdef.ini files was changed from FTS 40 to Agilent-TS5XX0. It is intended that users will modify this field to reflect their particular system name.

The default value of null (item 35) in the dsdef.ini and ssdef.ini was changed from 0 to 0.0 .

S. The profiler identifies the time consumption of different elements of a testplan. It is best run on "working" but not yet time optimized testplans. Profiling is automatically disabled for a testplan that is judged as failed because the testplan has likely terminated early without executing all of the tests in the testplan.

   Profiling a testplan that has failures can be done by setting the Ignore All Failures option to true. The result will be a testplan that runs to completion (even though tests will have failed) and profiling information will be available.

T. The standard format file for spreadsheet data logging (ssfmtdef.ini) does NOT include an entry that will enable the recording of report information. The following .ini file will result in report information being inserted into your spread sheet based data log records.

```
(null)   15    2 #This is a comma separated format for spread sheets     SCHEMA:2
ACTION        1 "" ""    ""    0
BATCH         0 "TEST","INDEX","STATUS","VALUE","NOMINAL","LOW_LIMIT","HIGH_LIMIT"\n
      ""    ""    -1  35
BLOCK         1 "" ""    ""    0
BOARD         1 "" ""    ""    0
DIGITAL       1 "" ""    ""    0
LIMIT         1 "" ""    ""    0
MEASUREMENT   1 "" \n    ,     7  9  49  51  55  34  31  25
NODELIST      1 "" ""    ""    0
OPEN          1 "" ""    ""    0
PHANTOM       1 "" ""    ""    0
REPAIR        1 "" ""    ""    0
REPORT        1 "" \n    ""    1 43
SHORTFROM     1 "" ""    ""    0
SHORTS        1 "" ""    ""    0
SHORTTO       1 "" ""    ""    0
```

   The only change between this .ini file and the standard one shipped with TxSL is the 13th line starting with the word report. The original file has all null entries, similar to the lines 12 and 14.

Section 11. CHANGES FROM VERSION 2.0 TO 3.0

A. The pass/fail status of each element in a measurement array is now being reported and logged individually.

B. The action definition editor was modified to edit old C Style actions.

   NOTE: The new editor does not allow the user to create a new action definition with this older style.

C. Early releases of Agilent TestExec SL supported an internal symbol table named TestplanGlobals for which support was dropped at release 2.1. Because old testplans sometimes use this symbol

table, whose scope is the entire testplan, support for it was reinstated to improve backward compatibility.

D. The OnIdlePoll callback was called only once while Agilent TestExec SL was idle. Now it is called continuously while Agilent TestExec SL is idle. This was a problem for implementing automation which requires polling of digital lines so the tester would know when the DUT was in place.

E. The SequenceLocals symbol table is now shared across testplan sequences by default.

   NOTE: This behavior was changed to maintain compatibility with some testplans created between the 2.0 & 2.1 releases that attempted to use the SequenceLocals symbol table to share symbols across testplan sequences. However, we STRONGLY recommend that you use TestPlanGlobals instead of SequenceLocals for sharing symbols across testplan sequences.

F. The System symbol table now contains two new symbols used to retain information about the most recent exception that occurred while running a testplan. The values of these symbols are valid only if the Exceptions check box is enabled on the Reporting tab in Agilent TestExec SL's Testplan Options box (Options | Testplan Options | Reporting).

   The new symbols are:

   UnhandledError - A string array that contains the contents of the exception stack when an exception was detected while running a testplan. The array's contents are the exception strings that appear in Agilent TestExec SL's Report window.

   UnhandledErrorSource - A string that contains the name of the test that was executing when the exception was detected. If no test was executing, this field is blank

G. The skip flags may now be set and saved with a testplan for all statement types - not just for tests.

H. The default behavior for all new testplans is halt after 1 test failure, previously the default behavior was to ignore all failures.

I. The DUT is now failed if one test fails within the testplan. The 'Halt on failures' feature no longer has any influence on the pass/fail value of a DUT. This feature only regulates the number of tests that may fail before sequencing of a testplan will halt. In previous releases the 'Halt on failures' feature would also be used in determining a DUT's pass/fail status -i.e. if this value was three and the number of failing tests was less than or equal to three then the DUT was considered to have passed. This is no longer true, if this value is set to three then the sequencer will stop running tests if there are more then three failures - however the DUT will be marked as passed ONLY if there are NO failed tests

within the testplan.

J. The report output now contains the testplan sequence exit status.

K. The sequencer has an exit code of SEQ_EXCEPTION_HALT even when the exception sequence is populated with tests, previously this exit code was returned only if the exception sequence was not populated with tests.

L. It is now possible to open a TxSL file (.tpa, .ust, .umd & .sym files) from Windows Explorer by simply double clicking the file.

   Note: Since the .sym extension is fairly popular the install will look to see if it is already being used by another application if it is then we don't change the file type association.

   You may also specify the ini file that you would like to use as a command line argument to TxSL. For example, running TxSL with the following command:
   tstexcsl yourfile.ini
   will start TxSL using the settings specified in yourfile.ini.

M. TxSL was not flushing the datalogging buffer when the sequence was halted from a paused state. Now the datalog buffer is always flushed when the sequence is halted.

N. TxSL now initializes the operator field of the system symbol table to "No Operator".

Agilent Technologies' Test and Measurement Support, Services, and Assistance
Agilent Technologies aims to maximize the value you receive, while minimizing your risk and problems. We strive to ensure that you get the test and measurement capabilities you paid for and obtain the support you need. Our extensive support resources and services can help you choose the right Agilent products for your applications and apply them successfully. Every instrument and system we sell has a global warranty. Support is available for at least five years beyond the production life of the product. Two concepts underlie Agilent's overall support policy: "Our Promise" and "Your Advantage."

Our Promise
Our Promise means your Agilent test and measurement equipment will meet its advertised performance and functionality. When you are choosing new equipment, we will help you with product information, including realistic performance specifications and practical recommendations from experienced test engineers. When you use Agilent equipment, we can verify that it works properly, help with product operation, and provide basic measurement assistance for the use of specified capabilities, while your instrument is under warranty or technical support contract. Many self-help tools are available on Agilent's web site.

Your Advantage
Your Advantage means that Agilent offers a wide range of additional expert test and measurement services, which you can purchase according to your unique technical and business needs. Solve problems efficiently and gain a competitive edge by contracting with us for calibration, extra-cost upgrades, out-of-warranty repairs, and on-site education and training, as well as design, system integration, project management, and other professional engineering services. Experienced Agilent engineers and technicians worldwide can help you maximize your productivity, optimize the return on investment of your Agilent instruments and systems, and obtain dependable measurement accuracy for the life of those products.

Agilent Email Updates

www.agilent.com/find/emailupdates
Get the latest information on the products and applications you select.

Agilent T&M Software and Connectivity
Agilent's Test and Measurement software and connectivity products, solutions and developer network allows you to take time out of connecting your instruments to your computer with tools based on PC standards, so you can focus on your tasks, not on your connections. Visit www.agilent.com/find/connectivity for more information.

By internet, phone, or fax, get assistance with all your test & measurement needs

Online assistance:
www.agilent.com/find/assist

Phone or Fax

United States:
(tel) 800 452 4844

Canada:
(tel) 877 894 4414
(fax) 905 282 6495

China:
(tel) 800 810 0189
(fax) 0800 650 0121

Europe:
(tel) (31 20) 547 2323
(fax) (31 20) 547 2390

Japan:
(tel) (81) 426 56 7832
(fax) (81) 426 56 7840

Korea:
(tel) (82 2) 2004 5004
(fax) (82 2) 2004 5115

Latin America:
(tel) 305 269 7500
(fax) 305 269 7599

Taiwan:
(tel) 080 004 7866
(fax) (886 2) 2545 6723

Other Asia Pacific Countries:
(tel) (65) 375 8100
(fax) (65) 836 0252
Email: tm_asia@agilent.com

Printed in the USA February 17, 2004

Agilent Technologies